

UNITED STATES DISTRICT COURT  
NORTHERN DISTRICT OF CALIFORNIA

TERADATA CORPORATION, et al.,  
Plaintiffs,  
v.  
SAP SE, et al.,  
Defendants.

Case No. [18-cv-03670-WHO](#)

**CLAIM CONSTRUCTION ORDER**

**BACKGROUND**

There are five patents asserted in this case. All the patents relate to computer-implemented database systems. Generally, all of them are related to data storage, organization, retrieval, analysis, or removal.

The '179 Patent (Patent Number 7,617,179, issued November 10, 2009), is titled "System and Methodology for Cost-Based Subquery Optimization Using a Left-Deep Tree Join Enumeration Algorithm," and teaches "query optimization" in a "relational database system." '179 Patent (Dkt No. 211-3). The claimed query optimizer identifies a plan containing "access methods" to obtain specific data from various tables. '179 Patent: 2:40-3:4. The optimizer considers whether to join or combine data within different tables and potential methods to accomplish this. *Id.* at 3:5-24. The optimizer then selects an "optimal access plan" with favorable execution costs. *Id.* at 5:46-49.

The '421 Patent (Patent Number 9,626,421, issued April 18, 2017), is titled "ETL-Less Zero-Redundancy System and Method for Reporting OLTP Data." '421 Patent (Dkt. No. 211-9). It relates to database systems, particularly transactional and reporting database systems, and teaches a system that allows for synchronization of data stored in row format and data stored in column format. *Id.* at 1:15-17, 2:30-32.

The '437 Patent (Patent Number 7,421,437, issued September 2, 2008), is titled "System and Method for a Data Dictionary Cache in a Distributed System." '437 Patent (Dkt. No. 211-12). It teaches a distributed system for the management of data dictionary information. '437 Patent at

1. The distributed system has three layers, a user layer, application layer, and data access layer. *Id.* at 1:48-55. The user layer allows for interaction between the distributed system and the user. *Id.* at 2:36-37. The application layer provides services to the user and accesses information from the data access layer. *Id.* at 1:49-52. The data access layer provides for data storage in “in one or more data dictionaries” for the distributed system. *Id.* at 3.

The ’321 Patent (Patent Number 8,214,321, issued July 3, 2012), titled “Systems and Methods for Data Processing,” teaches a method and system allowing for transactional data in a “database warehouse” to be stored, grouped, and analyzed. ’321 Patent: 7:10-8:53 (Dkt. No. 211-14).

The ’516 (Patent Number 7,437,516, issued October 14, 2008), is titled “Programming Models for Eviction Policies,” and deals with memory management, specifically, for a virtual machine’s local memory which is implemented in a “cache.” ’516 Patent at 1 (Dkt. No. 211-17). The ’516 Patent teaches a method for putting objects in the cache and determining which objects to remove from in cache. *Id.* at 10:45-48, 13:50-55.

The parties dispute nine claim terms from these five patents.<sup>1</sup>

### LEGAL STANDARD

Claim construction is a matter of law. *See Markman v. Westview Instruments, Inc.*, 517 U.S. 370, 372 (1996); *Vitronics Corp. v. Conceptronic, Inc.*, 90 F.3d 1576, 1582 (Fed. Cir. 1996). Terms contained in claims are “generally given their ordinary and customary meaning.” *Vitronics*, 90 F.3d at 1582. In determining the proper construction of a claim, a court begins with the intrinsic evidence of record, consisting of the claim language, the patent specification, and, if in evidence, the prosecution history. *Phillips v. AWH Corp.*, 415 F.3d 1303, 1313 (Fed. Cir. 2005); *see also Vitronics*, 90 F.3d at 1582. “A claim term used in multiple claims should be construed consistently . . . .” *Inverness Med. Switzerland GmbH v. Princeton Biomeditech Corp.*, 309 F.3d 1365, 1371 (Fed. Cir. 2002).

“The appropriate starting point [ ] is always with the language of the asserted claim itself.”

---

<sup>1</sup> As noted below, the parties disputed a 10th claim term, from the ’421 Patent, but agreed during claim construction briefing that its plain meaning sufficed.

1 *Comark Commc'ns, Inc. v. Harris Corp.*, 156 F.3d 1182, 1186 (Fed. Cir. 1998). “[T]he ordinary  
2 and customary meaning of a claim term is the meaning that the term would have to a person of  
3 ordinary skill in the art in question at the time of the invention, i.e., as of the effective filing date  
4 of the patent application.” *Phillips*, 415 F.3d at 1312. “There are only two exceptions to this  
5 general rule: 1) when a patentee sets out a definition and acts as his own lexicographer, or 2) when  
6 the patentee disavows the full scope of a claim term either in the specification or during  
7 prosecution.” *Thorner v. Sony Computer Entm’t Am. LLC*, 669 F.3d 1362, 1365 (Fed. Cir. 2012).

8 “Importantly, the person of ordinary skill in the art is deemed to read the claim term not  
9 only in the context of the particular claim in which the disputed term appears, but in the context of  
10 the entire patent, including the specification.” *Phillips*, 415 F.3d at 1313. “Claims speak to those  
11 skilled in the art,” but “[w]hen the meaning of words in a claim is in dispute, the specification and  
12 prosecution history can provide relevant information about the scope and meaning of the claim.”  
13 *Electro Med. Sys., S.A. v. Cooper Life Scis., Inc.*, 34 F.3d 1048, 1054 (Fed. Cir. 1994) (citations  
14 omitted). “[T]he specification is always highly relevant to the claim construction analysis.  
15 Usually, it is dispositive; it is the single best guide to the meaning of a disputed term.” *Vitronics*,  
16 90 F.3d at 1582. “However, claims are not to be interpreted by adding limitations appearing only  
17 in the specification.” *Id.* “Thus, although the specifications may well indicate that certain  
18 embodiments are preferred, particular embodiments appearing in a specification will not be read  
19 into the claims when the claim language is broader than such embodiments.” *Id.*

20 Finally, the court may consider the prosecution history of the patent, if in evidence.  
21 *Markman*, 52 F.3d at 980. The prosecution history may “inform the meaning of the claim  
22 language by demonstrating how the inventor understood the invention and whether the inventor  
23 limited the invention in the course of prosecution, making the claim scope narrower than it would  
24 otherwise be.” *Phillips*, 415 F.3d at 1317 (citing *Vitronics*, 90 F.3d at 1582-83); *see also Chimie*  
25 *v. PPG Indus., Inc.*, 402 F.3d 1371, 1384 (Fed. Cir. 2005) (“The purpose of consulting the  
26 prosecution history in construing a claim is to exclude any interpretation that was disclaimed  
27 during prosecution.”) (internal quotations omitted).

28 In most situations, analysis of this intrinsic evidence alone will resolve claim construction

disputes. *Vitronics*, 90 F.3d at 1583. However, “it is entirely appropriate . . . for a court to consult trustworthy extrinsic evidence to ensure that the claim construction it is tending to from the patent file is not inconsistent with clearly expressed, plainly apposite, and widely held understandings in the pertinent technical field.” *Pitney Bowes, Inc. v. Hewlett-Packard Co.*, 182 F.3d 1298, 1309 (Fed. Cir. 1999). Extrinsic evidence “consists of all evidence external to the patent and prosecution history, including expert and inventor testimony, dictionaries, and learned treatises.” *Markman*, 52 F.3d at 980. All extrinsic evidence should be evaluated in light of the intrinsic evidence, *Phillips*, 415 F.3d at 1319, and courts should not rely on extrinsic evidence in claim construction to contradict the meaning of claims discernible from examination of the claims, the written description, and the prosecution history, *Pitney Bowes*, 182 F.3d at 1308 (citing *Vitronics*, 90 F.3d at 1583). While extrinsic evidence may guide the meaning of a claim term, such evidence is less reliable than intrinsic evidence. *Phillips*, 415 F.3d at 1318-19.

## DISCUSSION

### I. '179 PATENT

#### A. “Optimal access plan” / “Optimize”

For example, as used in Claim 1 of the '179 Patent:

In a database system, a method for optimizing a database query for execution by a processor, the method comprising ... optimizing each query block to determine an **optimal access plan** for the query block based upon selecting pre-computed subquery access methods and join methods for subquery plan nodes of the query block as well as access methods, join methods, and join order for other plan nodes of the query block having favorable execution costs, wherein each query block is **optimized** without transformation of the subqueries using the pre-computed access methods and join methods;

'179 Patent at 38:47-48, 38:64-67-39:1-5.

Patent/Term	SAP's Proposal	Teradata's Proposal
<b>Optimal access plan:</b> Claims 1, 23 of the '179 Patent	“the estimated best plan, considering estimates of execution costs, among the plans considered”	“the access plan that has the lowest execution cost associated with it”

<b>Optimize[d]</b> : Claims 1, 23 of the '179 Patent	“find/found the estimated best plan, considering estimates of execution costs, among the plans considered”	“find the lowest execution cost”
--	--	----------------------------------

Both parties agree that the claims do not require finding the “absolute best plan,” but only a “reasonable plan,” and that the optimizer may only consider a subset of possible plans. Declaration of David Maier (“Maier Decl.,” Dkt. No. 211-1) ¶ 45; Teradata Responsive Claim Construction Brief (“TD CC,” Dkt. No. 219) at 6. In response to SAP’s position, Teradata states that it would be willing to add “among the plans considered” to its proposed definitions of “Optimal access plan” and “optimize[d].” TD CC 6. The real issue in dispute is whether “execution cost” is the *only* criterion used during the optimization process (Teradata’s position) or whether the patent simply requires “consideration” of the execution costs of various plans (SAP’s position).

Teradata relies on its expert, Dr. Jaideep Srivastava, who notes that SAP’s own extrinsic evidence (the Ramakrishnan textbook) supports Teradata’s definition. *See* Declaration of Jaideep Srivastava (Dkt. No. 219-1) ¶ 24. The textbook explains that optimizing involves “[e]stimating the cost of each enumerated plan and choosing the plan with the lowest estimated cost.” *See* Ramakrishnan, *et al.* “Database Management Systems,” McGraw Hill Press (Third Ed. 2003), Dkt. No. 211-5 at 479. Teradata also points to the text of the ’179 Patent, which identifies “execution cost” as the only criterion, noting that the optimization process:

determin[es] an optimal access plan for each query block based upon selecting access methods, join methods, and join order for plan nodes of the query optimization graph *having favorable execution costs*; and constructing a detailed access plan for execution of the database query based upon the optimal access plan determined for each query block.

’179 patent, 5:46-49 (emphasis added)); Srivastava Decl. ¶ 24.

SAP disagrees with limiting the optimization criteria to only “execution costs.” SAP Claim Construction Reply Brief (SAP Reply) 1. SAP points out that the Patent specifies considering “favorable execution costs,” but favorable does not mean lowest. *Id.* SAP also

contends that the Patent identifies other factors to be considered, pointing to one embodiment where plan selection is based on a “property vector” that is used to “compare the current access plan with the best plan that has been previously found.” Reply Declaration of David Maier (Dkt. 222-1) ¶ 5; ’179 Patent 18:58-60. SAP notes that the Patent goes on to explain that “[a] main component of the property vector is the cost estimate associated with a plan node or a partial access plan” which confirms other components are considered. SAP Reply 2 (quoting ’179 patent at 18:60-62).

Given the heavy and repeated emphasis on “costs” in the claim language and specification, cost is obviously a key if not primary factor in optimization. However, Teradata’s attempt to exclude from consideration any other non-cost factors is simply not supported.

**Adopted Construction – “Optimal access plan”: The estimated best plan, considering estimates of execution costs, among the plans considered.**

**Adopted Construction – “Optimize[d]”: Find/found the estimated best plan, considering estimates of execution costs, among the plans considered.**

**B. “Query optimization graph”**

For example, as used in Claim 1 of the ’179 Patent (emphasis added):

In a database system, a method for optimizing a database query for execution by a processor, the method comprising: receiving a database query including at least one subquery; building a **query optimization graph** for each query block of the database query, the **query optimization graph** including plan nodes representing subqueries of each query block;

’179 Patent at 38:47-53.

Patent/Term	SAP’s Proposal	Teradata’s Proposal
<b>Query optimization graph:</b> Claims 1,3, 23 of the ’179 Patent	“an internal representation of a query block or derived table block, structured as a graph”	“a representation of a query block in the form of a hypergraph (a graph in which an edge can join any number of vertices) that includes vertices (nodes) and edges (connections between two nodes), including hyperedges (a connection that can

		link more than two nodes) in which nodes can represent subplans and quantifiers; and in which each node representing subplans includes an array of access methods and an array of join methods”
--	--	---

Both sides admit that the preferred embodiment of the '179 Patent describes a “hypergraph.” Teradata relies heavily on the fact that in discussing the preferred embodiment in the “Detailed Description of a Preferred Embodiment” section, “Definition 4” defines the “query optimization graph” (QOG) *as* a hypergraph. '179 Patent, 14:47-49. Teradata argues, therefore, that the QOG must be defined as a hypergraph, noting that there are no other types of embodiments of QOGs disclosed in the '179 Patent.

SAP responds by pointing out that Definition 4 is not contained in the “glossary section” (which defines terms used throughout the Patent) but only in the definitions section of the preferred embodiment, which makes it improper to define for the Patent as a whole. SAP also argues that Teradata’s definition (including the mention of edges and nodes) is improper because it would make dependent claim 3 (which covers the “method of claim 1, wherein said building step includes building a query optimization graph including subplans and quantifiers of each query bloc”) superfluous. SAP Claim Construction Brief (“SAP CC,” Dkt. No. 211) 5; '179 Patent 39:11-14; *Phillips v. AWH Corp.*, 415 F.3d 1315 (Fed. Cir. 2005) (en banc) (“[T]he presence of a dependent claim that adds a particular limitation gives rise to a presumption that the limitation in question is not present in the independent claim.”). Teradata contends that this doctrine does not apply where an explicit definition is included in the patent itself and that SAP is impermissibly attempting to broaden the scope of the QOG term. TD CC 5. Teradata, however, never directly addresses the fact that Definition 4 is used to describe the preferred embodiment and not the scope of the invention itself.

SAP also argues the query optimization graph is not limited to hypergraphs by pointing to



intrinsic evidence including prior art references, examiner statements, and inventor's arguments. SAP CC 5. For instance, SAP points to two examples in the prior art cited by the examiner that depict QOGs that are not hypergraphs - a "tree structure" and a QOG that is an "internal representation of a complete SQL statement, possible composed of multiple 'subquery blocks.'" JCCPS Exhibit 1 [Dkt. No. 2016-1] at 3 (citing language from the Young-Lai Patent). SAP contends that these types of graphs appeared in the prior art cited by the examiners, and, as a result, a POSITA would have known that a QOG was not limited to a hypergraph.

I agree with SAP that Definition 4 is relevant to the preferred embodiment and not (as the glossary is) relevant to the Patent as a whole and, relatedly, that the use of a hypergraph in the preferred embodiment does not require QOGs to be hypergraphs.

**Adopted Construction – "Query optimization graph": An internal representation of a query block or derived table block, structured as a graph.**

### **C. Query Block**

As used in Claim 1:

In a database system, a method for optimizing a database query for execution by a processor, the method comprising: receiving a database query including at least one subquery; building a query optimization graph for each **query block** of the database query, the query optimization graph including plan nodes representing subqueries of each **query block**.

'179 Patent at 38:47-53.

Patent/Term	SAP's Proposal	Teradata's Proposal
<b>Query block</b> Claims 1, 2, 3, 23 of the '179 Patent  [all asserted claims]	"an atomic portion of a query that can be separately optimized"	"a part of a query that contains multiple parts because the query contains derived tables, views, and/or subqueries"

Both sides agree that a query block is the smallest part of a query that can be separately optimized. Teradata would consider including the word "atomic" in its definition if the parties could clarify the meaning of atomic since it does not hold a "well-known meaning for potential jurors." TD CC 8. In response to Teradata's concern over the word atomic, SAP proposes an alternative construction: a query block is "the smallest portion of a query that can be separately



1 optimized.” SAP 4.

2 The remaining dispute is whether the additional features of a query (containing the query  
3 block) identified in the Patent specification should be included in this definition. Both parties  
4 derive their definitions from the “glossary” section of the ’179 Patent which states, “[a] query  
5 block refers to an atomic portion or block of a query that has more than one block because the  
6 query contains derived tables views, and/or subqueries.” ’179 Patent: 4:43–45. The glossary  
7 definition then gives three examples of what query blocks can be.

8 SAP argues that its construction should be adopted because it explains what a query block  
9 is. SAP CC 8. SAP omits the “has more than one block because the query contains derived  
10 tables, views, and/or subqueries” part of the glossary definition, not because it is “inaccurate” but  
11 because it is “unnecessary” and difficult for a jury to understand. *Id.* Teradata contends that by  
12 omitting this language, SAP is attempting to omit the “very thing [it] has named query block” and  
13 is an attempt to expand the scope of the patent. TD CC 8 (internal quotations omitted).

14 SAP contends that Teradata’s definition, focusing on “why” something is a query block,  
15 could lead to juror confusion and could lead a juror to improperly understand that a query block  
16 encompasses any part of the query even if it is too small or too large to be optimized. SAP CC 8.  
17 SAP cites to Teradata’s expert’s admission that Teradata’s proposed definition only “explains  
18 why a query block would be created.” SAP Reply 4 (quoting Srivastava ¶ 26).

19 After considering both sides’ arguments, in my Tentative Opinion issued prior to the June  
20 12, 2020 Hearing, I suggested a construction of this claim term. Dkt. No. 265. Both sides agreed  
21 to my proposed construction. June 12, 2020 Transcript (Transcript) 20:18-24.

22 **Adopted Construction – “Query block”: A query block refers to the smallest portion**  
23 **of a query (which has more than one block) that can be separately optimized.**

## 24 **II. ’421 PATENT**

### 25 **A. “share a consistent view of said database information”**

26 For example, as used in Claim 1 of the ’421 Patent:

27 A computer system storing a computer program for processing  
28 database information for both transacting and reporting, said  
computer program being executed by said computer system, the

computer system comprising... in response to a database update request, said relational database management system component updates said database information stored in said row format, said relational database management system component notifies said column-oriented data processing component of said database update request, and said column-oriented data processing component updates said database information stored in said column format, whereby said relational database management system component and said column-oriented data processing component **share a consistent view of said database information**

'421 Patent at 19:64-67, 20:12-23.

Patent/Term	SAP's Proposal	Teradata's Proposal
<b>Share a consistent view of said database information:</b> Claims 1,19, 20 of the '421 Patent  [all asserted claims]	"update the row-format data (by the relational database management system component) and update the column-format data (by the column-oriented data processing component) within the same database transaction"	"database information that if accessed by the relational database management component is the same as if accessed by the column-oriented data processing component"

SAP argues that the Patent makes clear that the "column-format data and the row-format data are updated within the same database transaction." SAP CC 10; Maier Decl. ¶ 81. Dr. Maier contends that a POSITA would have understood "share a consistent view of said database information" to have multiple meanings in the art. Maier Decl. ¶ 81. However, based on the Patent and prosecution history, a POSITA would have understood it consistent with SAP's proposed construction, including the cabining of the process to "within the same transaction." *Id.* ¶¶ 81-84. SAP contends that Teradata's construction is insufficient because it could imply that the two components of data are updated in separate transaction that would be contrary to the term "share a consistent view of said database information." SAP CC 11.

Teradata asserts that SAP's construction does not require data consistency and hides this with the phrase "within the same database transaction." TD CC 16. Teradata's expert, Dr. Shahram Ghandeharizadeh states that a POSITA would have understood the term to refer to the "database information, not the database transaction, as being the same." Declaration of Shahram Ghandeharizadeh (Dkt. No. 219-3) ¶ 35. Teradata contends that SAP's definition fails to show the

significant limitation that the “two components are working with the same data at all times.” TD CC 15. It argues that SAP’s use of the phrase “within the same database transaction” is an attempt to insert a specific embodiment into the claims that ignores other embodiments. TD CC 16.

SAP replies that it does not improperly read “consistent” out of the claim since “share a consistent view” refers to the timing of the updates, which their definition encompasses. SAP Reply 6. It asserts that its construction addresses the timing of the updates in a way that demonstrates the row-format and column-format data are updated in the same way because they are responding to the same update request. SAP Reply 5. SAP argues that Teradata fails to do this – and itself fails to guarantee consistency – because Teradata’s construction would be satisfied if both row-format and column-format data were updated in the same way but in different requests. SAP Reply 6.

I agree with SAP that there is adequate support for requiring the consistency to be achieved during the “same transaction” and adopts its proposed construction.

**Adopted Construction – “Share a consistent view of said database information”:**  
**Update the row-format data (by the relational database management system component)**  
**and update the column-format data (by the column-oriented data processing component)**  
**within the same database transaction.**

**B. “wherein generating the query response accesses only one or more columns needed directly for generating the query response”**

As used in Claim 1 of the ’421 Patent:

A computer system storing a computer program for processing database information for both transacting and reporting, said computer program being executed by said computer system, the computer system comprising... in response to a query request to retrieve data, said column-oriented data processing component generates a query response based on said database information stored in said column format, **wherein generating the query response accesses only one or more columns needed directly for generating the query response.**

’421 Patent at 19:64-67, 20:12-23.

In their briefing, the parties agreed that this claim does not need to be construed and its plain meaning adopted. Therefore, no construction is necessary.

### III. '437 PATENT

The parties seek construction of only one claim term from the '437 Patent.

#### **“a plurality of data dictionary cache at an application level”**

As used in Claim 1:

A computer-implemented method comprising ...referencing a **plurality of data dictionary cache at an application level** to obtain the data type associated with the system information.... selecting a data dictionary cache from **the plurality of data dictionary cache at the application level**, if the data dictionary cache includes the data type

'437 Patent at 7:65, 8:8-10, 8:17-19.

Patent/Term	SAP's Proposal	Teradata's Proposal
<b>A plurality of data dictionary cache at an application level:</b> Claims: 1, 6, 11, 15 of the '437 Patent	“multiple data dictionary caches at a layer of software, <u>different from a data access layer</u> , that provides services to a user of data dictionary information, and obtains that data dictionary information from the data access layer”	“multiple data dictionary caches that are in corresponding application servers existing in a layer different from the data access level”

SAP added the underlined clause in response to Teradata's argument that SAP's definition confused the “application level” and the “data level.” TD CC 18; SAP CC Reply 8. Teradata takes no issue with that clarification. The parties still dispute whether there is a need to include “corresponding” from Teradata's definition, and whether it would confuse a jury into thinking that correspondence between two layers is required. SAP CC 14. SAP argues that the use of “corresponding” improperly implies that a correspondence between “data dictionary caches and application servers is required.” SAP CC 15. SAP and Maier argue that a POSITA would understand that such correspondence between the servers would not be required.

The next issue is the location of the data dictionary caches. Teradata asserts that it takes no position on “whether the claimed data dictionary caches must reside on one application server, or a plurality of application servers,” just that the “specification and prosecution history make clear

that any such caches must be at the application level (i.e. in one or more application servers) and not the data access level.” However, SAP’s expert, Maier contends that a POSITA would not have “inferred that there was a correspondence between a cache in the application layer and an application server” which he argues is confirmed in the prosecution history. Maier Decl. ¶¶ 92-93.

At the hearing, Teradata agreed that “correspondence” could be dropped from its definition, nullifying that dispute. However, it reasserted its position that its language focusing on *application servers* should remain in to avoid ambiguity. Transcript 46:24 - 47:6. I disagree. Teradata’s attempt to insert that additional limitation of a required connection to an “application server” into this construction is without support.

**Adopted Construction – “a plurality of data dictionary cache at an application level”:  
Multiple data dictionary caches at a layer of software, different from a data access layer, that provides services to a user of data dictionary information, and obtains that data dictionary information from the data access layer.**

#### **IV. ’321 PATENT**

Initially, the parties disagree over what the qualifications for a POSITA would be for the ’321 patent; Teradata contends through its expert Dr. Gandecharizadeh that a “higher” level of skill (a master’s degree or a bachelor’s in computer science plus four years’ experience) is appropriate for this patent. Ghandeharizadeh Decl. ¶¶ 17-20. Both parties agree that the qualifications of a person of skill in the art should not affect the claim construction analysis for any patent. TD CC 2; SAP CC 1.

##### **A. “Mapping” / “Mapping Table”**

As used in claim 4:

A data processing system comprising... a mapping table for **mapping** a sub-set of the set of online analytical processing cubes to the at least one class of online analytical processing cubes

’321 Patent at 7:46, 8: 5-7.

As used in claim 7:

The computer program product of claim 6, the online analytical processing cube being identified by a second **mapping table**.

*Id.* at 8:48-50.

Patent/Term	SAP's Proposal	Teradata's Proposal
<b>Mapping:</b> Claims 1, 4, 6 of the '321 Patent	"creating and storing, <u>in computer system memory or secondary storage for a computer system</u> , an association between two data elements in the computer system such that a computer can locate a data element using that association	"associating or assigning"
<b>Mapping table:</b> Claim 7 of the '321 Patent	"a computer-implemented data structure that holds associations or assignments, each between two data elements in a computer system"	"a computer-implemented data structure that holds associations or assignments"

The disputes on these terms are whether "mapping" must be stored and whether the mapping is limited to "between two data elements." The underlined text in SAP's proposal was offered in response to Teradata's argument that storage needed to be defined.

On the first issue, SAP argues that in all embodiments the mapping is stored, which makes sense because its purpose is to allow applications to access and process entities stored in tables and cubes assigned to classes, and it points to references in the Patent to mapping being "retrieved", not created anew. '321 Patent 5:45-54. Teradata replies that there is nothing in the Patent showing that there is a requirement that either the "mapping" or "mapping table" be stored "persistently" and contends that storage of mapping is simply permissive in the Patent. '321 Patent 6:11-17 (noting mapping and mapping table data "may be" stored). Teradata also accuses SAP of attempting to collapse the distinction between mapping and mapping table, because the Patent explains only that "corresponding mapping may be stored as a further mapping table." *Id.*

As to the second issue, SAP argues that in the patent, "mapping" assigns pairs of elements and a POSITA would have understood this to be true. SAP CC 18. Teradata disagrees with SAP's construction that the assignments are "pairwise"; since multiple tables can be assigned to

any class, there is not necessarily a 1:1 relationship. TD CC 11. SAP reasserts that although more than one table can be mapped to a class, the assignment of each table to class is done in a pairwise way one table at a time and that Teradata's expert's statements also support this construction.

SAP Reply 10-11.

I agree with SAP that mapping is stored given the logical need and evidence in the specification. But I agree with Teradata that there is insufficient basis to limit that mapping to "pairwise" elements as a matter of claim construction.

**Adopted Construction – "Mapping": Creating and storing, in computer system memory or secondary storage for a computer system, an association between data elements in the computer system such that a computer can locate a data element using that association.**

**Adopted Construction – "Mapping Table": A computer-implemented data structure that holds associations or assignments.**

#### **B. "Online Analytical Processing Cube"**

As used in Claim 1:

A data processing method comprising... providing a set of **online analytical processing cubes** in a data warehouse, each **online analytical processing cube** specifying a layout for transactional data storage;'321 Patent at 7:12, 16-18.

Patent/Term	SAP's Proposal	Teradata's Proposal
<b>Online analytical processing cube:</b> Claims 1, 2, 4, 6, 7, 8 of the '321 Patent	"a data structure (or a computer-executable definition thereof) to store multidimensional data, where data to be stored in the data structure is or will be provided by online analytical processing"	"a data structure designed to store multidimensional data, where data to be stored in the data structure is provided by online analytical processing."

The parties agree that the OLAP cube is a "a data structure designed to store



1 multidimensional data,” referring to a physical instance of a data structure.<sup>2</sup> In dispute is whether  
2 the construction should also include the data-structure “definition” used to create that instance.

3 SAP argues that the “definition” term should be included because the ’321 Patent contains  
4 several statements indicating that OLAP cubes may be “predefined,” and therefore a POSITA  
5 would have understood OLAP cubes in context of the Patent to include both the data structure and  
6 definition. SAP CC 18; SAP Reply at 11 (“in instantiated OLAP cube is created from its  
7 definition, making the definition and instance two stages of the same cube.”). SAP’s expert agrees  
8 that a POSITA would have understood the term to reference both the data structure and the  
9 definition of the data structure. Maier Decl. ¶¶ 102-103.

10 Teradata responds that SAP is seeking to expand the claimed term because the “existence  
11 of a definition does not necessarily mean an instance of it exists.” Ghandeharizadeh Decl. ¶ 31.  
12 The definition of an OLAP cube, according to Teradata, must be *implemented* in a data structure in  
13 order to embody what is claimed, and the patent cannot cover only the definition without at least  
14 one instance.

15 SAP replies that it is not saying that the definition of a data structure and an instantiated  
16 cube are the same, but that in context of the Patent a POSITA would have understood an OLAP  
17 cube to potentially refer to either. Therefore, it amended its proposed construction, to include the  
18 underlined text. But I agree with Teradata that SAP is attempting to broaden this term without  
19 support and will adopt Teradata’s proposal.

20 **Adopted Construction – “Online Analytical Processing Cube”: A data structure**  
21 **designed to store multidimensional data, where data to be stored in the data structure is**  
22 **provided by online analytical processing.**

23 **C. “[means for] invoking an online analytical processing component to fill the online**  
24 **analytical processing cubes with transactional data”**

25 As used in Claim 1:

26 A data processing method comprising ...invoking an online analytical  
27

28 <sup>2</sup> OLAP cubes are data structures that are the “basis for transaction data storage in prior art data warehouse systems.” ’321 Patent 1:26-27.

processing component to fill the online analytical processing cubes with transactional data[.]

'321 Patent at 7:12, 29-31.

As used as “means-plus-function” element in Claim 4: A data processing system comprising ... means for invoking an online analytical processing component to fill the online analytical processing cubes with transactional data[.]

*Id.* at 7:46-8:8-10.

Patent/Term	SAP's Proposal	Teradata's Proposal
[means for] invoking an online analytical processing component to fill the online analytical processing cubes with transactional data: Claim 1, 4 of the '321 Patent [all asserted claims]	“using software to transfer transactional data into online analytical processing cubes”	“using an online analytical processing component to transfer transactional data into empty online analytical processing cubes”
	This claim element is governed by 35 U.S.C. 112(6). <b>Structure/Material/Acts</b> An application program.	This claim element is governed by 35 U.S.C. 112(6). <b>Corresponding Structure:</b> none; indefinite

The parties disagree about the meaning of the term when invoked as a step as depicted in Claim 1 and regarding the use of the term as a “means-plus-function” element as depicted in Claim 4. Regarding the use of the term as a “means-plus-function” element, the parties argue whether the specification adequately describes a corresponding structure for performing the function.

### 1. “Invoking Step”

First, the parties disagree whether the OLAP cubes must be empty when “transfer[ing] transactional data into” them. SAP argues that there is nothing in the claims nor the specification requiring that the OLAP cubes *must* be empty. Rather, the language implies only that they “may be” empty. SAP CC 20. Teradata responds that the statement that the OLAP cubes “may be empty” is the only evidence cited by SAP, and that this supports its position that the cubes must initially always be empty. TD CC 13. Teradata also asserts that the claims and specifications show that the claims follow an ordered set of steps and based on that ordering, when the

“invoking” step occurs the cube has not been filled. *Id.* SAP replies that nothing in the claims requires the steps to be performed in order, and there is nothing in the claims that require the OLAP cubes to be empty when provided.

I agree with SAP. Teradata’s proposed construction inserting “empty” is not supported.

**Adopted Construction – “Invoking an online analytical processing component to fill the online analytical processing cubes with transactional data”: Using software to transfer transactional data into online analytical processing cubes.**

## 2. “Means-Plus-Function”

Turning to the question of whether there is an adequately disclosed structure to perform the claimed means, SAP argues that the specification discloses that the “invoking” function is used with an application program, which is a sufficiently disclosed structure. SAP CC 19. SAP lists several examples of application programs that a POSITA would have known could invoke OLAP functionality. *Id.* at 19-20. It relies on *Zeroclick, LLC v. Apple Inc.*, where the Federal Circuit found that the claims were not invalid for indefiniteness under § 112 ¶ 6. 891 F.3d 1003, 1008 (Fed. Cir. 2018) (finding that in the patent “program” and “user interface code” were not used as “generic terms” but “as specific references to conventional graphical user interface programs or code, existing in prior art at the time of the inventions.”).

Teradata argues that a generic application program is not enough to satisfy the structure requirements under the Federal Circuit case law. It relies, instead, on *Aristocrat Techs. Australia PTY Ltd. v. Int’l Game Tech.*, where the Federal Circuit affirmed the district court’s ruling that providing a “general purpose microprocessor” was not a sufficient disclosure of structure under 112 ¶ 6 as the patent holder did not “disclose particular structure in the specification ... to avoid pure functional claiming.” 521 F.3d 1328, 1333 (Fed. Cir. 2008). Teradata argues since the claim does not explain how the software works, it does not satisfy § 112 ¶ 6.

SAP replies that here the particular structure is an application program, which is not a prohibited simple reference to “general-purpose computer processor” or just “appropriate programming.” It contends that *Zeroclick* itself demonstrated that a computer program can be an adequately disclosed structure. SAP Reply 12. Finally, it cites to a recently decided Federal

Circuit opinion that clarified that *Aristocrat's* holding requires a “specific algorithm” only when the asserted structure is a reference to a “general-purpose computer or microprocessor”, which is not what this Patent discloses. *Nevro Corp. v. Boston Sci. Corp.*, 2020 WL 1802794 \*1, \*5 (Fed. Cir. April 9, 2020).

At the hearing, both sides admitted that the determination of whether there is a sufficiently disclosed structure for this means does not need to be determined on claim construction. Both agreed that this determination could be deferred until summary judgment when this and other means-plus-function claims will be addressed. Transcript at 39:6-12 (Teradata); 43:3-7 (SAP). I will therefore address all of Teradata’s means-plus-function challenges at that time.

## V. '516 PATENT

The parties dispute three related phrases from the '516 Patent: “eviction policy plug-in” / “storage policy plug-in” / “plug-in”

As used in Claim 1 of the '516 Patent:

A method, comprising ...configuring respective first and second cache portions by plugging said first and second **eviction policy plug-ins** into a cache management software program and plugging respective first and second **storage plug-ins** into said cache management software program;

'516 Patent at 19:25, 43-47.

Patent/Term	SAP's Proposal	Teradata's Proposal
<b>Plug-in:</b> Claims 1, 9, 17 of the '516 Patent [all asserted claims]	“piece of software or code that provides a requisite functionality”	“a discrete body of code that is separate from a main program and can be added to and removed from the main program without modification to the main program, such that the discrete body of code adds functionalities to the main program when added to it”
<b>Eviction policy plug-in:</b> Claims 1, 9, 17 of the '516 Patent	“the actual piece of software or code that dictates the removal of an object from cache”	“plug-in for performing a sorting method and an eviction timing method.”

<b>Storage policy plug-in (or storage plugin):</b> Claims 1, 9, 17 of the '516 Patent	“the actual piece of software or code that executes the “get” and “put” operations for objects stored in cache”	“plug-in for performing cache storage treatment of stored data.”
--	---	--

SAP argues that Teradata’s construction would include limitations that are not supported by intrinsic evidence. These limitations would require the plug-ins to be:

1) a *discrete* body of code; 2) *separate* from a main program; 3) able to be *added* to *and removed* from the main program *without modification* to the main program; and 4) *add functionalities*.

SAP Reply 13.

Teradata contends that these limitations are well-supported by the general understanding of the “known term of art” of a “plug-in” and that because SAP did not act as its own lexicographer and did not otherwise define “plug-in” to be something other than what was generally understood at the time, its construction should be adopted. Declaration of Dr. Daniel Menascé (Dkt. No. 219-5) ¶¶ 24-25 (“eviction policy plug-in” and “storage policy plug-in” contain a known term of art, “plug-in,” a POSITA would have understood what these terms mean in context of the known term). In turn, Teradata argues that the definition of “plug-in” cannot be separated from the definition of “eviction policy plug-in” or “storage policy plug-in.” TD CC 21.

SAP responds that eviction policy plug-ins and storage policy plug-ins were not used in this Patent consistent with existing terms of the art and SAP’s definitions should be adopted because they are taken from the specification. SAP CC 23; '516 Patent, at 7:8–11 (explaining an embodiment of the eviction policy plug-in); '516 Patent, at 7:1–4 (explaining an embodiment of the storage policy plug-in); Maier Decl. ¶¶ 106-08. SAP argues that Teradata improperly ignores the intrinsic evidence and construes “plug-in” in isolation, which is improper because “plug-in” even has several different terms of the art which could lead to jury confusion. SAP CC 23. SAP admits that a “plug-in” is a common term of art but it argues the term has no “single well-accepted meaning” and that Teradata’s extrinsic evidence supports this by showing “inconsistent meanings of ‘plug-in.’” SAP Reply 15. Unlike other examples of plug-in and as required by Teradata’s definition, SAP asserts that the plug-ins do not simply add functionality but are “required” for the

1 patent's functionality. SAP CC 23; Maier Decl. ¶ 110 (stating "[a] POSITA would have  
2 recognized that the 'eviction policy plug-in' and 'storage plug-in' performed specific functions  
3 that were required for the operation of the cache manager of the '516 patent").

4 Finally, SAP contends that Teradata's proposal adds limitations that are not supported by  
5 either the cited intrinsic or extrinsic evidence: namely, "none of the cited evidence indicates that a  
6 'plug-in' must be 'a discrete body of code,' or requires that a 'plug-in' 'can be added to and  
7 removed from the main program without modification to the main program.'" SAP CC 24.  
8 Teradata replies that the claim language shows that a plug-in must be "plug[ed] into a main  
9 program." TD CC 24 (internal quotations omitted). Further, the specification shows "that a plug-  
10 in in a library is a discrete body of code that is plugged into a cache manager... without modifying  
11 the rest of the main program"). '517 Patent, Figures 4, 16. SAP agrees with Teradata that the  
12 claims use of "plugging" shows that the plug-in is added to the "cache management software," but  
13 figures in the '516 patent show the "eviction policy plug-in and the storage plug-in within the  
14 cache manager ... indicating that the two plug-ins are neither discrete nor separate." *See* Figures 4-  
15 8 (showing the plug-ins within the cache management software).

16 In my Tentative Order, I proposed the following definition of plug-in: a "piece of software  
17 or code that can be added to and removed from the main program without modification to the  
18 main program to provide a requisite functionality." Dkt. No. 265 at 4. I also tentatively adopted  
19 SAP's proposed definitions for eviction and storage policy plug-ins. *Id.* As the hearing, Teradata  
20 was happy with the Court's construction of plug-in, but questioned how it would fit into the  
21 policies. Transcript at 30:10-13. SAP agreed with part of the suggested definition but took issue  
22 with "removed," as removal of the plug-in is not required or supported by the claim language or  
23 specification. Transcript at 28:21-25. SAP also argues that the "without modification" language  
24 needs clarification so that it is clear that the addition of the plug-in would be done "without  
25 modification to the other functionality of the main program." Transcript at 29:12-16.

26 I adopt SAP's proposed revision of my proposed language. My construction does not  
27 foreclose Teradata's argument that the plug-in must be a "discrete" modular set of code and that  
28 the claim language does not cover a developer's subsequent rewrite of a program to add

functionality. Those arguments can be raised on summary judgment.

**Adopted Construction – “Plug-in”:** A piece of software or code that can be added to the main program without modification to the other functionality of the main program.”

**Adopted Construction – “Eviction policy plug-in”:** The plug-in that dictates the removal of an object from cache.

**Adopted Construction – “Storage policy plug-in (or storage plugin)”:** The plug-in that executes the “get” and “put” operations for objects stored in cache.

### CONCLUSION

For the foregoing reasons the disputes terms are construed as follows:

**“Optimal access plan”:** The estimated best plan, considering estimates of execution costs, among the plans considered.

**“Optimize[d]”:** Find/found the estimated best plan, considering estimates of execution costs, among the plans considered.

**“Query optimization graph”:** An internal representation of a query block or derived table block, structured as a graph.

**“Query block”:** A query block refers to the smallest portion of a query (which has more than one block) that can be separately optimized.

**“Share a consistent view of said database information”:** Update the row-format data (by the relational database management system component) and update the column-format data (by the column-oriented data processing component) within the same database transaction.

**“A plurality of data dictionary cache at an application level”:** Multiple data dictionary caches at a layer of software, different from a data access layer, that provides services to a user of data dictionary information, and obtains that data dictionary information from the data access layer.

**“Mapping”:** Creating and storing, in computer system memory or secondary storage for a computer system, an association between data elements in the computer system such that a computer can locate a data element using that association.



1           **“Mapping Table”:** A computer-implemented data structure that holds associations  
2 or assignments.

3           **“Online Analytical Processing Cube”:** A data structure designed to store  
4 multidimensional data, where data to be stored in the data structure is provided by online  
5 analytical processing.

6           **“Invoking an online analytical processing component to fill the online analytical  
7 processing cubes with transactional data”:** Using software to transfer transactional data  
8 into online analytical processing cubes.

9           **“Plug-in”:** A piece of software or code that can be added to the main program  
10 without modification to the other functionality of the main program.”

11           **“Eviction policy plug-in”:** The plug-in that dictates the removal of an object from  
12 cache.

13           **“Storage policy plug-in (or storage plugin)”:** The plug-in that executes the “get” and  
14 “put” operations for objects stored in cache.

15           **IT IS SO ORDERED.**

16 Dated: July 15, 2020

17  
18 

19 William H. Orrick  
20 United States District Judge  
21  
22  
23  
24  
25  
26  
27  
28